Embed, Track and Authenticate Images Online with SDW-WebCrawler

Embed, track and authenticate images online with SDW-WebCrawler

Pratheepan Yogarajah, Joan Condell, Kevin Curran and Paul Mc Kevitt
School of Computing and Intelligent Systems,
University of Ulster, Northern Ireland, U.K.

Abbas Cheddad
Umea Centre for Molecular Medicine (UCMM),
Umea Universitet,
901 87 Umea, Sweden.

***Abstract*—The Internet is a widely open source to everyone to access Web pages. Using a web browser anyone can access websites. Because of this facility people can easily download images from websites without the owner's knowledge and use them in their own documents. Also image content may be modified for illegal purposes. Therefore a system is needed to authenticate images over the Web. Web image authentication is a challenging task that requires web crawlers to track and download images for authentication. Most of the known web image tracking engines such as TinEye and PicScout retrieve images according to the image infringement of the original image. However, these systems do not have the facility to authenticate the retrieved image, i.e. whether the retrieved image is similar to the original image or any image content alteration has occurred in the retrieved image and who is the copyrighted owner of the retrieved image.**

In order to solve the above mentioned drawbacks this paper presents a framework to protect image content, track it over the internet and authenticate the content. The proposed framework is based on self-embedding (i.e. where secret data and a binary version of the image are encrypted and embedded into the image), tracking (i.e. where a web crawler traverses over the internet to download images) and self-authentication (i.e. where the binary version of the hidden data is extracted to authenticate the image). Also another advantage of the proposed system is that it does not need the original image for the authentication process.

I. Introduction

The World Wide Web is a system of interlinked hypertext documents accessed via the Internet. With a web browser, one can view web pages that may contain text, images, videos, and other multimedia and navigate between them via hyperlinks. Once owners post their images on the Internet, any visitor can save those pictures or videos and it would be so easy for them to post those pictures onto their own websites and then take credit for them. Also they can forge the original content and make fake imagery for any illegal use.

Image forgery is a technique for generating a fake image by altering, combining, or creating new image content. Using commonly available software, it is now feasible to perform digital image editing by inserting or deleting objects from those images without introducing visible artifacts. The ease of editing visual data in the digital domain has facilitated unauthorized tampering without leaving any perceptible trace.

Therefore the main questions of the owner of the image are "how do I know my image has been used in an illegal manner on the internet?" and "who is using it without permission from me?" As a answer, a system is needed to protect the image, track the image over the internet to find out who is using it and authenticate the image to find out whether the image content has been modified or not. There are some products available in the market such as TinEye and PicSout and those give a partial solution for the above mentioned questions. TinEye [1] is a reverse image on-line based search engine developed and offered by Ide, Inc. It searches the web for user submitted images to locate infringement on their copyright. This makes TinEye a potential tool for the copyright holders to track their images over the internet.

Fig. 1. (a) submitted lena image. (b) a retrieved image from internet using TinEye tool. (c) a retrieved image along with the corresponding information of current user of that image.

Another tool is called PicScout Image Tracker [2] and it is also a web-based program that allows a user to upload images. PicScout then starts the tracking process of the uploaded image. Whenever a match to an uploaded image is found the user receives a report on that match from PicScout. Both systems works under similar concepts:

- fingerprint user submitted image
- search the Web to find images using web crawler over different websites
- compare the images that are similar to the submitted image and send matches to the user

The main goal of methods mentioned above is that if a user submitted an image then these systems find similar images even though those images have been changed in terms of e.g.

intensity, rotated, translated, cropped or stretched. Figure 1(b) shows one of the retrieved images for the submitted 'Lena' image, see Figure 1(a), using TinEye on-line tool. Figure 1(c) shows the result of who is using the retrieved image. PicSout also works similar to the TinEye on-line tool. The main drawbacks of these two systems are:

- need original image for search
- could not automatically verify the image to find which portion of the image content has been altered
- without original image the owner could not prove his/her ownership

To overcome the drawbacks of the above systems, we propose a system that protects images using a self-embedding steganography based watermarking technique, track the image over the internet using our web crawler and verify the images using our authentication method. The main goal of our work is to prevent the possibility of creating a forgery that goes undetected. Manipulations on images fall into different categories such as the intensity changes, noise, removing and replacing content.

This paper is mainly focused on the removing and replacing content type of attack. To detect the unauthorised manipulation of such image footages, an image authentication system should verify that the image taken has not been tampered with in the spatial domain. Authentication has always been an important issue [12]. Content authentication is a process by which a user is guaranteed that image content is original and has not been maliciously modified.

Based on the embedding of specific digital signatures (a set of authentication bits), the image may be viewed by authorized personnel but made unavailable to others. Self-embedding refers to the process where a compressed copy of the image is embedded into itself during watermarking. After self-embedding, it is possible to recover portions of the image that have been altered without accessing the original image.

This paper is organised as follows. Our method is discussed in Section II. The secure image encryption algorithm is explained in section III. Section IV explains our embedding algorithm. The web crawling is explained in section V. The sections VI and VII explain results and provide a discussion and conclusions respectively.

II. Our Method

Our method contains two main concepts called self authentication and web crawler. Our self authentication method consists of two parts, self-embedding and authentication. In the self-embedding phase, a binary version of the original content (also called payload) is embedded into the original image (also called the cover image). Since we need a means of protecting the image files against forgery, it is essential that the payload will carry as much information from the cover image as possible. An approximation of the cover image file can be achieved by applying binarization techniques which result in a binary image demanding only 1 bit per pixel for storage.

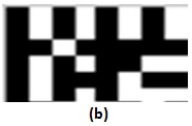
Figure 2 shows three different binary versions (b), (c) and (d) of the original image (a). Figure 2 (b), (c) and (d) are created by dithering, thresholding and Canny edge detection respectively. All these binary versions can be considered as spatial payload for self-embedding. Pratheepan et al. [3] use the above three binary versions of payloads for self-embedding. In the authentication phase, they argue that extracted thresholded payload shows much more "visible" original content than extracted edge and dither payloads. They only use the Human Visual Systems (HVS) to compare the original payload content against the extracted payload content. Instead of HVS, we are interested on recovering the approximation of the original image content to compare with the original image content for image content authentication.

Fig. 2. (a) original image. (b) created by dithering using the method described by Floyd and Steinberg [7], (c) created by thresholding the original image and (d) created by Canny edge operator respectively.

Cheddad et al. [4] argued that the dithered version of payload is a better choice to recover the approximation of the original image. Therefore we use the dithered version of the original image as our payload. There exist different algorithms to generate an inverse halftone image. Neelamani et al. [5] propose an inverse halftoning in the Wavelets domain using Floyd-Steinberg [7] kernels to generate the error diffusion signal. We use their Wavelet-based Inverse Halftoning via De-convolution (WInHD) to recover the approximation of the original signal. This recovery of the original image is important as our system does not need the original image for authentication.

ON	U	0	1	0	1	0	1	0	1
	U	0	1	0	1	0	1	0	1
		0	0	1	0	0	0	0	0
	S	0	1	0	1	0	0	1	1
	D	0	1	0	0	0	1	0	0
	W	0	1	0	1	0	1	1	1

(a)



(b)

Fig. 3. (a) The binary representation of the 6 characters. (b) corresponding binary image.

Further, each image should have it's owner name as this information allows the owner to prove his/her ownership. Therefore we considered using the owner name (ON) (i.e.

authorised body of the image) for embedding. In total 6 characters are allocated to represent ON for each image. Each character is represented as an 8 bits binary representation. For example, if we considered a ON, "**UU SDW**", is converted to an 6x8 bits binary representation, see Figure 3(a). Then the calculated 6x8 binary matrix is directly converted to a binary image, see Figure 3(b). Then the calculated binary images are embedded in the original image (cover image) based on the highly secure self-embedding algorithm proposed in [4].

III. Image Encryption Algorithm

This algorithm is explained based on [4] and the encryption algorithm is fully described in [6]. A hash function is more formally defined as the mapping of bit strings of an arbitrary finite length to strings of fixed length [8]. Here we attempt to extend SHA-1 (the terminology and functions used as building blocks to form SHA-1 are described in the US Secure Hash Algorithm 1, [9]) to encrypt digital 2D data. The introduction of Fast Fourier Transform (FFT) forms together with the output of SHA-1 a strong image encryption setting. Let the key bit stream be $\lambda_{k,l}$ where the subscripts k and l denote the width and height after resizing the keys bit stream respectively, i.e., 8, $M * N$, where M, N are the plain image's dimension.

The FFT will operate on the Discrete Cosine Transform (DCT) of $\lambda_{k,l}$ subject to Eq. 2.

$$f(u,v) = \frac{1}{N} \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} F(x,y) e^{-2\pi i(xu+yv)/N} \qquad (1)$$

where $F(x,y) = DCT(\lambda_{k,l})$ satisfying Eq. (2). Note that for the transformation at the FFT and DCT levels we do not utilise all of the coefficients. Rather, we impose the following rule, which generates at the end a binary random-like map. Given the output of Eq. 1 we can derive the binary map straightforwardly as:

$$Map(x,y) = \begin{cases} 1 & iff \quad f(u,v) > 0 \\ 0 & otherwise \end{cases} \qquad (2)$$

This map takes the positive coefficients of the imaginary part to form the ON pixels in the map. Since the coefficients are omitted the reconstruction of the password phrase is impossible, hence the name Irreversible Fast Fourier Transform (IrFFT). In other words, it is a one way function which accepts initially a user password. This map finally is XORed with the binary version of each colour component separately. Another phenomenon that we noticed and we would like to exploit is the sensitivity of the spread of the FFT coefficients to changes in the spatial domain.

Therefore if we couple this with the sensitivity of the SHA-1 algorithm to changes of the initial condition, i.e., password phrase, we can easily meet the Shannon law requirements, i.e., confusion and diffusion. For instance a small change in the password string will, with overwhelming probability, result in a completely different hash and thus a different image by extension. So, the core idea here is to transform these changes into the spatial domain where we can apply 2D-DCT and 2D-FFT that introduce the aforementioned sensitivity to the two dimensional space. As such, images can be easily encoded securely with password protection. Next section explains the embedding algorithm.

IV. The Embedding Algorithm

The payloads are securely embedded using the encryption algorithm explained above. Let **C** and **P** be the cover image and the payload respectively. The stego-image **S** can be obtained by the following embedding procedure:

A. Embedding Procedure

- Step 1: Encrypt **P** using the proposed encryption method to find **P'**.
- Step 2: Transform **C** to YCbCr color space.
- Step 3: Decompose the channel **Y** by one level of 2D-Discrete Wavelet Transform (DTW) to yield four subimages (CA, CH, CV, CD).
- Step 4: Convert the integer part of coefficients of CA into Binary Reflected Gray Code code (BRGC) and store the decimal values.
- Step 5: The coefficient's 3^{rd} least significant bit is chosen to embed the secret bit while random bits are embedded simultaneously into the coefficient's 1^{st} and 2^{nd} least significant bit. This procedure is known as masking and it helps overcome few compression errors.
- Step 6: Convert the modified BRGC code back to coefficients, restore the decimal precision and reconstruct Y'.
- Step 7: Convert Y'CbCr to RGB colour space and obtain the stego-image, i.e., **S**.

There are two payloads need to be embedded in cover image, owner name and spatial content. Therefore the **Y** is divided as two parts, header and body, see Figure 4. The owner name is embedded in the Header part. Here the size of the owner name is fixed, 6x8 bits. So the size of the header part is allocated 4xcolumn size. Therefore the size of body part is allocated (rows-4)xcolumn size. The spatial content, the dithered version of the particular image, is only embedded in the body part of that image. The embedded image is called the Stego image. Each Stego image contains two parts, the Header and the Body.

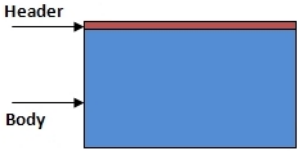


Fig. 4. Header and Body representation of the stego images.

An image is protected using the above mentioned self-embedding process. In the extraction procedure, WInHD is applied on the extracted binary image to recover the approximation of the original image. To track the protected images in the internet we developed a web crawler. Our web crawler is discussed in the next section.

V. Web Crawling

A web crawler is software used to traverse the internet and retrieve content of the individual pages passed by along the way. Following is the process by which web crawler works:

1) visit the web page
2) Pass through the visited page and retrieve all the links.
3) for each link retrieved, repeat the process

In the first step, a web crawler takes a URL and visits the page from the internet at the given URL. In the second step, a web crawler parses through the visited page and retrieves the links to the other pages. Each link in the page is defined with an HTML anchor tag. Web crawlers may assess the value of a URL or a content word by examining the HTML tag context in which it resides. For this, a crawler may need to utilize the tag tree of the HTML page [10], [11].

```
<html>
<head>
<title>Projects</title>
</head>
<body>
<h4>Projects</h4>
<ul>
  <li> <a href="blink.html">LAMP</a> Linkage analysis with multiple processors.</li>
  <li> <a href="nice.html">NICE</a> The network infrastructure for combinatorial exploration.</li>
  <li> <a href="amass.html">AMASS</a> A DNA sequence assembly algorithm.</li>
  <li> <a href="dali.html">DALI</a> A distributed, adaptive, first-order logic theorem prover.</li>
</ul>
</body>
</html>
```

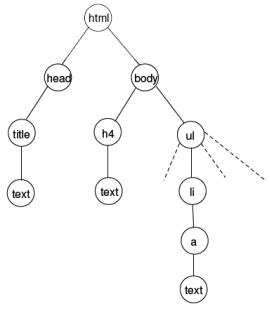


Fig. 5. An HTML page and the corresponding tag tree.

Figure 5 shows a tag tree corresponding to an HTML source. The <html> tag forms the root of the tree and various tags and texts form nodes of the tree. Unfortunately, many Web pages contain badly written HTML. For example, a start tag may not have an end tag (it may not be required by the HTML specification), or the tags may not be properly nested. In many cases, the <html> tag or the <body> tag is altogether missing from the HTML page. Thus structure-based criteria often require the prior step of converting a "dirty" HTML document into a well-formed one, a process that is called tidying an HTML page[1]. This includes both insertion of missing tags and the reordering of tags in the page. Tidying an HTML page is necessary for mapping the content of a page onto a tree structure with integrity, where each node has a single parent.

After the crawler has retrieved the links from the page, each link is added to the list of the links to be crawled. The third step of web crawling repeats the process. All the crawlers work in the recursive or loop fashion, but there are two different ways to handle it. Links can be crawled in a depth-first or breadth-first manner. Depth-first crawling each possible path to its conclusion before another path is tried. It works by finding the first link on the first page. It then crawls the page associated with the link, finding the first link in the new page, and so on, until the end of the path has been reached. The process continues until all the links have been exhausted.

In our method, the web crawling process is conducted by a single web crawler run on a personal computer. The web crawler performs a depth-first search for image links on potential websites. When the web crawler receives a URL to crawl as input, it looks for all potential URLs in the page. However since it does a depth first search, it recursively keeps crawling down every link that it traverses. A depth value is used as a cutoff point to stop the crawling process. Here the cutoff point three is defined as depth value. At this point, it has a collection of all links that it just crawled. Figure 6, describes the basic components of the proposed web crawler.

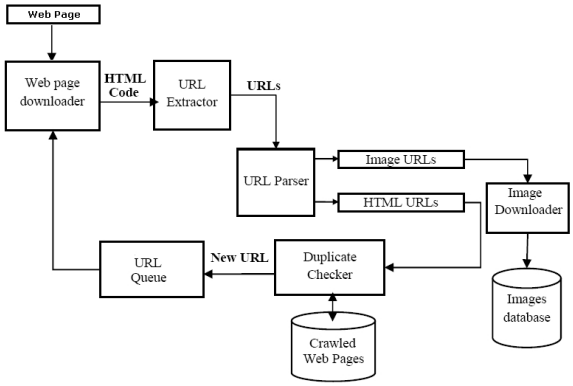


Fig. 6. Web Crawling.

The proposed web crawler consists of these components:

1) Web Page Downloader: This component reads a list of URLs and makes HTTP requests to get those web pages.
2) URL Extractor: This component is responsible for extracting URLs from a visited web page.
3) URL Parser: this component parses the URLs, pass the image URL to image downloader to be downloaded and pass the HTML URL to the duplicate checker.
4) Duplicate Checker: this component prevents the same URL being crawled again by checking the list of URLs that have already been crawled. If the URL page is not crawled yet, the duplicate checker adds it back to the URL Queue which is a list of URLs of Web pages that will be crawled.
5) Image Downloader: download and store the image and information about image such as URL of the image site and image type in the images database.

When an image is downloaded from the Web using our web crawler the authentication process is applied to authenticate the image. Next section shows some experimental results of our proposed system.

[1]http://www.w3.org/People/Raggett/tidy/ - accessed on 01/05/2011

VI. Results and Discussion

In this paper we discuss the development of a system to embed, track and authenticate images. When a image is downloaded the header part is tested for owner name. If the owner name is extracted and identified then the body part is considered for spatial content verification to find the modified content. We created a web page "http://www.infm.ulst.ac.uk/∼pratheepany/IMVIP2011" and inserted three images to test our system.

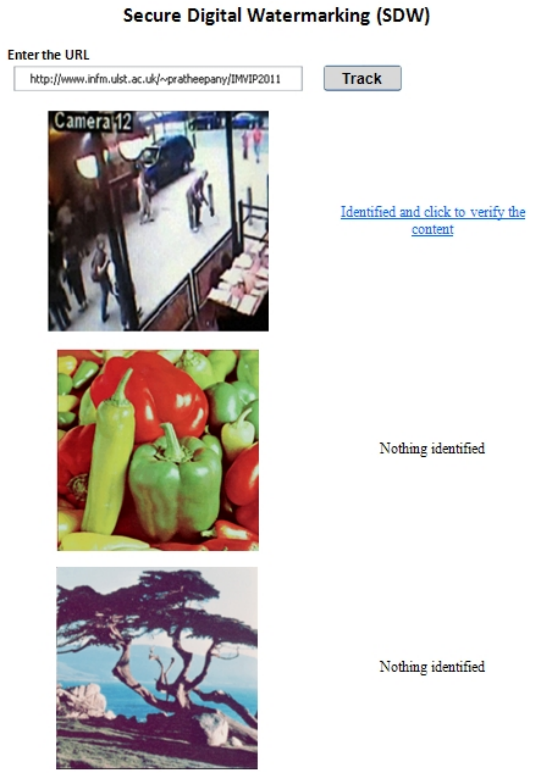


Fig. 7. Track images over the web using URLs.

Fig. 8. (a) found image from the Web. (b) extracted hidden content and reconstructed using WInHD. (c) altered region is shown with red color.

Figure 7 shows the results received for the given URL "http://www.infm.ulst.ac.uk/∼pratheepany/IMVIP2011". Our web crawler retrieved all three images from the given website. Then the authentication method considers retrieved images one by one and first checks the header part for the owner name "UU SDW". If owner name is extracted and identified from the image then "*Identified and click to verify the content*" message will be displayed next to the retrieved image else "*Nothing identified*" will be displayed.

Fig. 9. (a) Original image, (b) halftoned binary image, (c) and (d) are recovered images from JPEG 95% and 85% quality compression attacks respectively.

If "*Identified and click to verify the content*" link is clicked then the dithered binary version of the spatial image content will be extracted and the original image will be recovered using WInHD. Based on the recovered spatial content from the retrieved image and the spatial content of the retrieved image the modified region is identified using some image processing operations, see Figure 8.

The content of watermarked digital images can be easily attacked by using image processing operations such as lossy compression. Invisible watermarking requires a reasonable robustness against compression attacks. Lossy compression algorithms tend to remove invisible information that can be related to the watermark. Watermark robustness under image compression is an essential issue for image content protection. Therefore, watermarks should combine invisibility and robustness simultaneously.

Therefore lossy compression methods, such as JPEG, Block Truncation Coding (BLC) and Singular Value Decomposition (SVD), are applied to test the robustness of our proposed method against these compression attacks. **Block Truncation Coding (BTC)** is a lossy image compression technique. It divides the original images into blocks and then uses a quantizer to reduce the number of grey levels in each block while maintaining the same mean and standard deviation [13]. **Singular Value Decomposition (SVD)** is one of the most useful tools of linear algebra. It is a factorization and approximation technique which effectively reduces any matrix into a smaller invertible and square matrix. Using (SVD) for image compression can be a very useful tool to save storage space [14], [15].

We illustrate and evaluate the performance of the proposed method against JPEG, BTC and SVD compression attacks on grayscale images. Here we present experimental results using the image 'Lena' (256x256 pixels, grayscale). The 'Lena' image is shown in Figure 9(a).

Then the watermarked images are generated. These watermarked images are attacked by JPEG, BTC and SVD compression techniques. The binary watermarks are extracted from attacked watermarked images and the approximation of the original image is recovered, see Figure 9(c) and (d).

For quantitative evaluation, the PSNR (Peak Signal-to-Noise Ratio) is introduced to evaluate the performance between the original image and recovered image. The PSNR is defined as follows:

$$\mathbf{PSNR} = 10log_{10}(\frac{255^2}{\mathbf{MSE}})dB \tag{3}$$

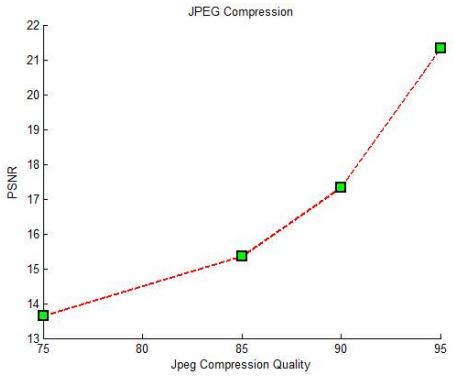


Fig. 10. results of JPEG compression attack on 'Lena' image.

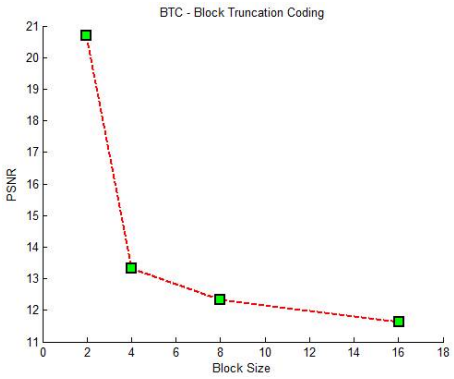


Fig. 11. results of BTC compression attack on 'Lena' image.

$$\mathbf{MSE} = \sum_{i=1}^{n}\sum_{j=1}^{m}\frac{(a_{i,j} - b_{i,j})^2}{n * m} \tag{4}$$

where $m*n$ is the image size, $a_{i,j}$ and $b_{i,j}$ are the corresponding pixel values of cover and recovered images.

Figure 10 shows the performance measure of JPEG compression quality factors 75%, 85%, 90% and 95%. Figure 11 shows the performance measure of BTC compression technique using block sizes 2x2, 4x4, 8x8 and 16x16. The performance measure of SVD compression attacks using singular values of 60 to 80 can be seen from Figure 12.

Our experimental results show evidence that the original content of the digital image can be recovered to a certain extent even though the watermarked image is attacked by lossy compression such as JPEG, BTC and SVD. Partial results of compression attacks performance measure appears in [16]

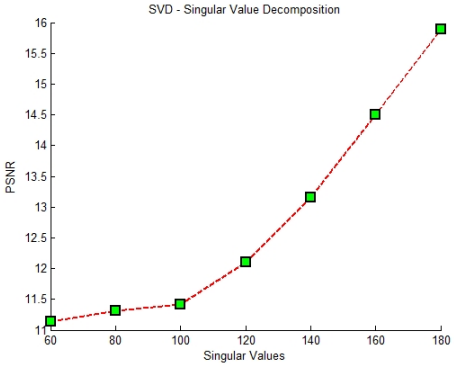


Fig. 12. results of SVD compression attack on 'Lena' image.

VII. Conclusion

We have proposed a novel approach to authenticate the images that appear on the Web. Spatial content of the original image and owner name (i.e. secret information) are encrypted and embedded using a highly secure information hiding technique, [4]. A web crawler is developed and used to retrieve the images from Web. Then an authentication method is applied to authenticate the originality of the image content. Our system can easily find which portion of the image has been altered and it can also recover the original content from the altered image as well. This paper presented a complete system to protect, track and authorise images appearing on the Web.

We only experimented the robustness of our proposed method against compression attacks, such as JPEG, BTC and SVD. Future work will involve to test our method against other attacks such as scaling, contrast adjustment, gamma correction and histogram equalisation.

References

[1] http://www.tineye.com/ - accessed 11/05/2011.
[2] http://www.picscout.com/products-services/imagetracker.html - accessed 11/05/2011
[3] Y. Pratheepan and J.V. Condell and K. Curran and A. Cheddad and P. McKevitt, *Video Authentication: A Self Embedding Steganography Approach*. Proceedings of The Irish Machine Vision and Image Processing Conference (IMVIP), 2010.
[4] A. Cheddad and J. Condell and K. Curran and P. McKevitt, *A secure and improved self-embedding algorithm to combat digital document forgery*, Signal Process., 89(12), pages 2324–2332, 2009.
[5] R. Neelamani and R. Nowak and R. Baraniuk, *Model-Based Inverse Halftoning with Wavelet-Vaguelette Deconvolution*, Proceedings of International Conference on Image Processing, pages 973–976, 2000.
[6] A. Cheddad and J. Condell and K. Curran and P. McKevitt, *A Hash-based Image Encryption Algorithm*, Optics Communications, 283(6), pages 879–893, 2010.
[7] R.W. Floyd and L. Steinberg, *An Adaptive Algorithm for Spatial Gray Scale*. Int. Symposium Digest of Technical Papers, Society for Information Displays, pages 36, 1975.
[8] W. Yong and L. Xiaofeng and D. Xiao and W. Kwok-Wo, *One-way hash function construction based on 2D coupled map lattices*, Inf. Sci., 178(5), pages 1391–1406, 2008.
[9] RFC3174 - US Secure Hash Algorithm 1 (SHA1), *http://www.faqs.org/rfcs/rfc3174.html*, 2001.
[10] G. Pant and F. Menczer, *Topical crawling for business intelligence*. In Proc. 7th European Conference on Research and Advanced Technology for Digital Libraries (ECDL 2003), Trondheim, Norway, 2003.
[11] G. Pant. *Deriving Link-context from HTML Tag Tree*. In 8th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery, 2003.
[12] F. Hartung and M. Kutter, *Multimedia watermarking techniques*. PIEEE, 87(7), pages 1079–1107, 1999.
[13] Chanda B, Dutta Majumder D. *Digital Image Processing and Analysis*. Prentice-Hall, 2000.
[14] Richards D, Abrahamsen A. *Image compression using singular value decomposition*. linear algebra applications, 2001.
[15] Prasantha H S, Shashidhara H L, Balasubramanya Murthy K N. *Image Compression using SVD*. Proc. of International Conference on Computational Intelligence and Multimedia Applications. 143–145, 2007.
[16] Pratheepan, Y., Condell, J.V., Curran, K., Cheddad, A., Mc Kevitt, P. *An Improved Self-Embedding Algorithm: A Robust Protection against Lossy Compression Attacks in Digital Image Watermarking*, In International Journal of Image Processing and Communications. 15-1:47-59, 2010.